

TWOFOLD VIDEO HASHING WITH AUTOMATIC SYNCHRONIZATION

Mu Li, Vishal Monga

Department of Electrical Engineering, The Pennsylvania State University, USA.

ABSTRACT

Video hashing finds a wide array of applications in content authentication, robust retrieval and anti-piracy search. While much of the existing research has focused on extracting robust and secure content descriptors, a significant open challenge still remains: Most existing video hashing methods are fallible to temporal desynchronization. That is, when the query video results by deleting or inserting some frames from the reference video, most existing methods assume the positions of the deleted (or inserted) frames are either perfectly known or reliably estimated. This assumption may be okay under typical transcoding and frame-rate changes but is highly inappropriate in adversarial scenarios such as anti-piracy video search. For example, an illegal uploader will try to bypass the ‘piracy check’ mechanism of YouTube/Dailymotion etc by performing a cleverly designed non-uniform resampling of the video. We present a new solution based on dynamic time warping (DTW), which can implement automatic synchronization and can be used together with existing video hashing methods. The second contribution of this paper is to propose a new robust feature extraction method called flow hashing (FH), based on frame averaging and optical flow descriptors. Finally, a fusion mechanism called distance boosting is proposed to combine the information extracted by DTW and FH. Experiments on real video collections show that such a hash extraction and comparison enables unprecedented robustness under both spatial and temporal attacks.

1. INTRODUCTION

Video hashing is a dimensionality reduction technique which transforms a raw video to a compact vector that can facilitate content-based retrieval. Other applications include video authentication, anti-piracy search and augmented reality. Robustness against content-preserving distortions is a central requirement of video hashing, and security applications often also require cryptographic key based randomization.

Existing literature and motivations of this paper: The existing video hashing techniques can be roughly classified into two types. The first type, lower order information methods, extract hash vectors directly from the video frames. The typical methods of this type include Radial hASHing (RASH, [1]), Discrete Cosine Transform (DCT, [2]), Centroids of

Gradient Orientations (CGO, [3]) and Temporally Informative Representative Images (TIRI, [4]). These methods either extract some geometric information such as RASH (which samples each frame using a set of lines centered at the frame’s midpoint) and CGO (which computes local gradients), or calculate some transform coefficients (usually discrete cosine transform due to its energy compaction property) like DCT and TIRI. Note that TIRI proposed frame averaging during hash extraction and showed this operation is very robust to temporal distortions. Although lower order information gained initial success, researchers discovered that higher order information can achieve even better performance. Methods of this type extract hash vectors not directly from video frames, but from correlations between nearby frames. One representative method in this type is HOOF [5, 6], where Histogram of Orientations of Optical Flow is used to extract hash vectors. Other advances include the use of multiple hash vectors to generate binary hash bits using spectral hashing [7, 8]. A practical challenge with [7, 8] is that as sufficient number of new videos are added to the database, retraining is needed and all hash vectors must be regenerated. Our goal is instead to develop fusion techniques such that model retraining does not influence existing hashes in the database. The central challenge we seek to overcome is the open problem of temporal desynchronization in video hashing. There have indeed been notable attempts in this direction, namely in [9, 10] where frame based image hashes can be used to synchronize audio or video. But these techniques invariably require complicated combinatorial optimization problem and are hence quite expensive. Further, the strategy to normalize the query video to the same length with reference video even after finding correspondence does not seem unique.

Contributions of this paper: This paper develops a new video hashing paradigm called: Twofold hashing. First, a preprocessing method based on DTW is implemented, which can automatically detect the positions of deleted /inserted frames quickly and reliably synchronize the distorted query video to the same length with the original reference video. Note that the proposed preprocessing is universal in the sense that it can be used together with virtually any robust feature extractor in existing video hashing methods. Next, we propose a new robust feature extractor called flow hashing (FH), which tries to blend the frame averaging operation in [4] with HOOF in [6]. Finally, we propose a fusion technique called

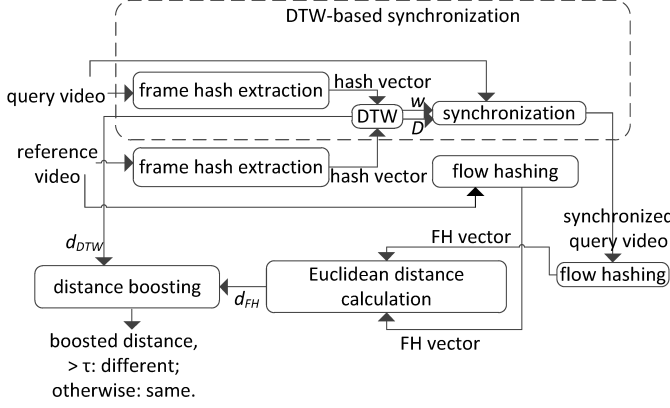


Fig. 1. Proposed twofold video hashing system.

distance boosting which aims to combine (fuse) the measure given by DTW-based preprocessing and Euclidean distance between FH hashes. Experiments confirm that the detection performance in terms of Receiver Operating Characteristics (ROCs), is significantly improved when compared against state of the art video hashing techniques.

2. ALGORITHM FORMULATION

2.1. DTW-based video synchronization

The central idea of the proposed synchronization is: Extract hash vectors from each frame, model the resulting hash vector as time series, and apply dynamic time warping to synchronize query time series to reference time series. Dynamic time warping [11] is based on dynamic programming principle [12], whose central idea is: Given the starting point, the problem of finding the optimal path to an end point is equivalent to first go to an optimal middle point and then find the optimal path starting from that middle point.

Specifically, we model two frame-based hash vectors \mathbf{h}_r^f (from reference video \mathcal{V}_r) and \mathbf{h}_q^f (from query video \mathcal{V}_q) as time series, apply DTW to compute an optimal warping path w and a distance d_{DTW} simultaneously. Here w represents an correspondence between \mathbf{h}_r^f and \mathbf{h}_q^f . A warping path is denoted by

$$w = \{(i_k, j_k)\}_{k=1}^p, \quad (1)$$

which represents i_k 'th frame in the query video corresponds to j_k 'th frame in the reference video. An example warping path is shown in Fig. 2, where the red points (point whose x -coordinate and y -coordinate are increased by 1 simultaneously comparing with the previous point on warping path) are of particular interest since they represent the beginning points of matching intervals that are separated by black dashed lines. DTW solves the following problem:

$$d_{DTW} \doteq \min_w \left(\sum_{k=1}^p D(i_k, j_k) \right), \quad (2)$$

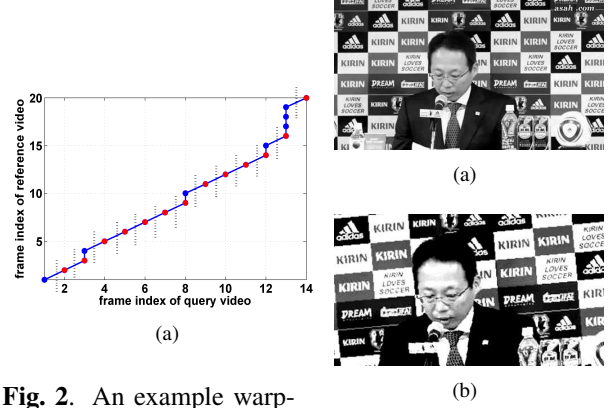


Fig. 2. An example warping path.

Fig. 3. A typical frame: (a) Original; (b) Attacked/Distorted.

where $D(\cdot, \cdot)$ is the basic metric (defined in Step 6 of Algorithm 1 in our scenario). Solving (2) is unfortunately combinatorially explosive; the standard DTW uses some constraints such as monotonicity and continuity to reduce size of search space. As a result, (3) is used to approximate (2):

$$\gamma(i, j) = \underbrace{D(i, j) + \min}_{\text{find the optimal present step}} \left(\underbrace{[\gamma(i-1, j), \gamma(i-1, j-1), \gamma(i, j-1)]}_{\text{warping path is optimal until the previous step}} \right). \quad (3)$$

The warping path w is computed by the index of the minimizers in (3) backwardly, and $d_{DTW} \approx \gamma(\frac{\text{length}(\mathbf{h}_r^f)}{2}, \frac{\text{length}(\mathbf{h}_q^f)}{2})$. Note that d_{DTW} is not a “metric” strictly because it allows two different points to have zero distance and does not satisfy triangle inequality. But d_{DTW} can measure how close two vectors are in our scenario. DTW has long been known as one of the most powerful ways of comparing time-series of different lengths. Further, recent algorithmic advances ensure that DTWs can be computed very fast [13]. We leverage these merits into developing a video synchronization method as stated in Algorithm 1.

2.2. Robust representation: Flow hashing (FH)

The central idea of flow hashing is to blend the frame averaging operation in [4] and HOOF feature in [5]. Frame averaging operation is shown to be robust to temporal attacks and can also help to reduce hash length [4]; HOOF is used because the pixel motions, as a higher order information, is one of the most definitive features of a video. Specifically, an optical flow is a vector field, which is a map X from a manifold

Algorithm 1 DTW-based video synchronization

- 1: **[Frame hash extraction]**: Apply 2-D DCT to frames of query video \mathcal{V}_q , extract the first horizontal and vertical coefficients (adjacent to DC) to form \mathbf{h}_q^f . The corresponding \mathbf{h}_r^f for reference \mathcal{V}_r was computed offline.
 - 2: **{%comment: Given \mathbf{h}_r^f and \mathbf{h}_q^f , Step 3 -Step 7 apply DTW to compute w , D and d_{DTW} .%}**
 - 3: **{%comment: Compute ℓ_2 distance between every two frames from \mathcal{V}_r and \mathcal{V}_q .%}**
 - 4: **for** each $n \in \{1, 2, \dots, \frac{\text{length}(\mathbf{h}_q^f)}{2}\}$ **do**
 - 5: **for** each $m \in \{1, 2, \dots, \frac{\text{length}(\mathbf{h}_r^f)}{2}\}$ **do**
 - 6: $D(n, m) = \|\mathbf{h}_q^f(1 + (n - 1) \cdot 2 : 2 \cdot n) - \mathbf{h}_r^f(1 + (m - 1) \cdot 2 : 2 \cdot m)\|_2$.
 - 7: **[Standard DTW]**: Given D , apply Equ. (3) to get optimal warping path w and distance d_{DTW} .
 - 8: **{%comment: Given w and D , Step 9 -Step 16 synchronize \mathcal{V}_q to the same length as \mathcal{V}_r , the synchronized video is saved in $\mathcal{V}_q^{\text{syn}}$.%}**
 - 9: **[Extract beginning points]**: Given w , extract the coordinates of the T matching intervals' beginning points (red points in Fig. 2), $\{x_i^1, y_i^1\}_{i=1}^T$, where x_i^1 is a frame index in \mathcal{V}_q , y_i^1 is a frame index in \mathcal{V}_r .
 - 10: **[Extract matching intervals]**: From beginning points, extract coordinates of other points in the same matching interval (blues points between the same two dashed lines with each red point in Fig. 2), $\{\{x_i^j, y_i^j\}_{j=1}^{B_i}\}_{i=1}^T$, B_i is the number of points in i 'th interval.
 - 11: **for** each $i \in \{1, 2, \dots, T\}$ **do**
 - 12: $\text{Mat}(i, :) = [x_i^p, y_i^p]$ where $D(x_i^p, y_i^p)$ is minimal among $\{D(x_i^j, y_i^j)\}_{j=1}^{B_i}$. **{%comment: $\text{Mat}()$ stores the coordinates of matching points; from each interval, choose matching point to be (x_i^p, y_i^p) such that $D(x_i^p, y_i^p)$ is smallest in the same interval.%}**
 - 13: **for** each $i \in \{1, 2, \dots, T\}$ **do**
 - 14: $\mathcal{V}_q^{\text{syn}}(:, :, \text{Mat}(i, 2)) = \mathcal{V}_q(:, :, \text{Mat}(i, 1))$.
 - 15: **[Interpolation]**: Apply interpolation if there are missing frames in the middle of $\mathcal{V}_q^{\text{syn}}$.
 - 16: **[Extrapolation]**: Apply extrapolation if there are missing frames in the beginning or end of $\mathcal{V}_q^{\text{syn}}$.
-

Algorithm 2 flow hashing

- 1: **[Frame averaging]**: Given the input video \mathcal{V}_{in} ,
 - 2: **for** each $j \in \{1, 2, \dots, \frac{\text{length}(\mathcal{V}_{in})}{J}\}$ **do**
 - 3: $\text{TIRI}(:, :, j) = \frac{1}{J} \sum_{k=1+(j-1) \cdot J}^{j \cdot J} \mathcal{V}_{in}(:, :, k)$.
{%comment: Frame averaging; it can also be done in overlapping segments.}
 - 4: **[Compute optical flow]**: Calculate optical flow between every two successive TIRI , and the histogram of orientations of the optical flow, counted by motion amplitude. Concatenate all histograms to form vector \mathbf{h}_{in}^o .
 - 5: **[Normalization]**: $\mathbf{h}_{in}^o = \frac{\mathbf{h}_{in}^o}{\|\mathbf{h}_{in}^o\|_2}$.
-

M to its tangent bundle TM ,

$$X : \underbrace{M}_{\text{a manifold}} \rightarrow \underbrace{TM}_{\text{tangent bundle of the manifold}}. \quad (4)$$

In our algorithm, M is \mathbb{R}^2 and we use [14] to compute optical flow X ; and the resulted optical flow is encoded through histogram of orientations counted by motion amplitude. The formal steps of FH are described in Algorithm 2.

2.3. Distance boosting

We propose distance boosting, which tries to fuse different distances through linear combination as AdaBoost [15] does for classifiers. Both frame-based hashes \mathbf{h}^f and flow hashes \mathbf{h}^o are utilized so that the overall detection performance can be improved. In the training phase:

$$d_{\text{DTW}}(\mathcal{V}_r, \mathcal{V}_q) \doteq \gamma\left(\frac{\text{length}(\mathbf{h}_r^f)}{2}, \frac{\text{length}(\mathbf{h}_q^f)}{2}\right), \quad (5)$$

$$d_{\text{FH}}(\mathcal{V}_r, \mathcal{V}_q) \doteq \|\mathbf{h}_r^o - \mathbf{h}_q^o\|_2, \quad (6)$$

$$d_{\text{boost}}(\mathcal{V}_r, \mathcal{V}_q) \doteq \alpha_1 \cdot d_{\text{DTW}}(\mathcal{V}_r, \mathcal{V}_q) + \alpha_2 \cdot d_{\text{FH}}(\mathcal{V}_r, \mathcal{V}_q), \quad (7)$$

we try to make d_{boost} between visually same videos smaller than that between visually different videos, as much as possible. Slack variable technique [16] used in 1-norm soft margin SVM is used here to realize this goal, i.e., we solve

$$\begin{aligned} \min_{\alpha_1, \alpha_2} \quad & \sum_{(i,j,k) \in \mathcal{J}} \left[\left(\alpha_1 \cdot d_{\text{DTW}}(\mathbf{h}_i^f, \mathbf{h}_j^f) + \alpha_2 \cdot d_{\text{FH}}(\mathbf{h}_i^o, \mathbf{h}_j^o) \right) \right. \\ & \left. + 1 - \left(\alpha_1 \cdot d_{\text{DTW}}(\mathbf{h}_i^f, \mathbf{h}_k^f) + \alpha_2 \cdot d_{\text{FH}}(\mathbf{h}_i^o, \mathbf{h}_k^o) \right) \right]_+ \\ \text{s.t.} \quad & \alpha_1 \geq 0, \alpha_2 \geq 0. \end{aligned} \quad (8)$$

$$\mathcal{J} \doteq \{(i, j, k) : \mathcal{V}_i \text{ and } \mathcal{V}_j \text{ are visually same videos,}$$

$$\mathcal{V}_i \text{ and } \mathcal{V}_k \text{ are visually different videos.}\} \quad (9)$$

Note that since (8) is convex (essentially equivalent to a linear program), fast numerical techniques yield the optimal solution. Note that distance boosting is inspired by the distance combination technique proposed in [17], which is based on distance metric learning. The differences from [17] are: 1) We use a different objective function which, unlike [17], does not need cross-validation; 2) in distance boosting, d_{DTW} is fused, which is not a metric. Further in comparison with spectral hashing techniques [7, 8], distance boosting only needs to update α_1 and α_2 rather than having to regenerate all hash vectors under retraining.

3. EXPERIMENTS AND ANALYSIS

Detection performance is measured by ROCs of a binary hypothesis test problem where H_1 assumes the query video \mathcal{V}_q

is a distorted version of the reference video \mathcal{V}_r ; H_0 assumes \mathcal{V}_q and \mathcal{V}_r are visually different videos. The error probabilities are defined as

$$P_M(\tau) = \Pr(d(\mathbf{H}(\mathcal{V}), \mathbf{H}(A(\mathcal{V}))) \geq \tau) \quad (10)$$

$$P_{FA}(\tau) = \Pr(d(\mathbf{H}(\mathcal{V}), \mathbf{H}(A(\mathcal{V}')) < \tau) \quad (11)$$

where $A(\cdot)$ denotes content-preserving attacks. The distortions/attacks we test against are: 1.) Spatial attack: Rotate 5 deg, crop to $[\frac{3}{4}\text{Width}, \frac{3}{4}\text{Height}]$, intensity changes from $[0.2, 0.8]$ to $[0, 1]$; (The visual effect is shown in Fig. 3) 2.) Temporal attack: 30% of frames are dropped randomly and non-uniformly; 3.) Spatio-temporal attack: Spatial attack plus Temporal attack as articulated above. We compare our algorithm with two widely cited methods in CGO [3] and TIRI [4]; the algorithm parameters are set so that FH, CGO and TIRI will produce hash vectors of roughly the same length (64, 80, 72, respectively). 700 videos are downloaded from YouTube. In each simulation, 700 matching video pairs and 700 visually distinct video pairs are used. In FH with distance boosting, 350 pairs are used in training and other 350 pairs are used in testing. Each video is normalized to $64 \times 64 \times 2$ f/s.

Benefits of automatic synchronization: We choose flow hashing as $\mathbf{H}(\cdot)$, and Euclidean metric as $d(\cdot, \cdot)$, and plot the ROCs of three cases: 1) DTW: Synchronized using the proposed method; 2) optimal: Assume the frame deletion/insertion positions are perfectly known; 3) random: Assume frames are deleted/inserted at random positions. Note that Case 2 though unrealistic provides a bound to benchmark given methods since it leads to perfect synchronization; Case 3 represents the current real-world scenario since YouTube for example doesn't know the deletion/insertion positions chosen by a potentially malicious uploader. Fig. 4 reveals that the proposed method can significantly improve detection performance under temporal attacks in particular.

Why distance boosting works: We make different choices of $d(\cdot, \cdot)$ to explain why the proposed distance boosting can improve detection performance. We plot the histogram of normalized d_{DTW} in (5), d_{FH} in (6) and d_{boost} in (7), respectively, in Fig. 5, from which we can see that if we only use either frame-based hashes \mathbf{h}^f (from which d_{DTW} is calculated), or flow hashes \mathbf{h}^o (from which d_{FH} is calculated), the histogram of distances between nonmatching pairs (red part in Fig. 5(a) and Fig. 5(b)) will have significant overlap with that between matching pairs (blue part in Fig. 5(a) and Fig. 5(b)). But if we fuse \mathbf{h}^o and \mathbf{h}^f using the proposed distance boosting method in (8), d_{boost} between matching video pairs will tend to be much smaller than that between nonmatching video pairs, resulting that the blue histogram and red histogram in Fig. 5(c) will be much less overlapping than if use only \mathbf{h}^o or \mathbf{h}^f . This reduced overlap in turn improves detection performance, which is verified next.

ROC comparisons against existing techniques: Finally, we synchronize the query video using the proposed DTW-based

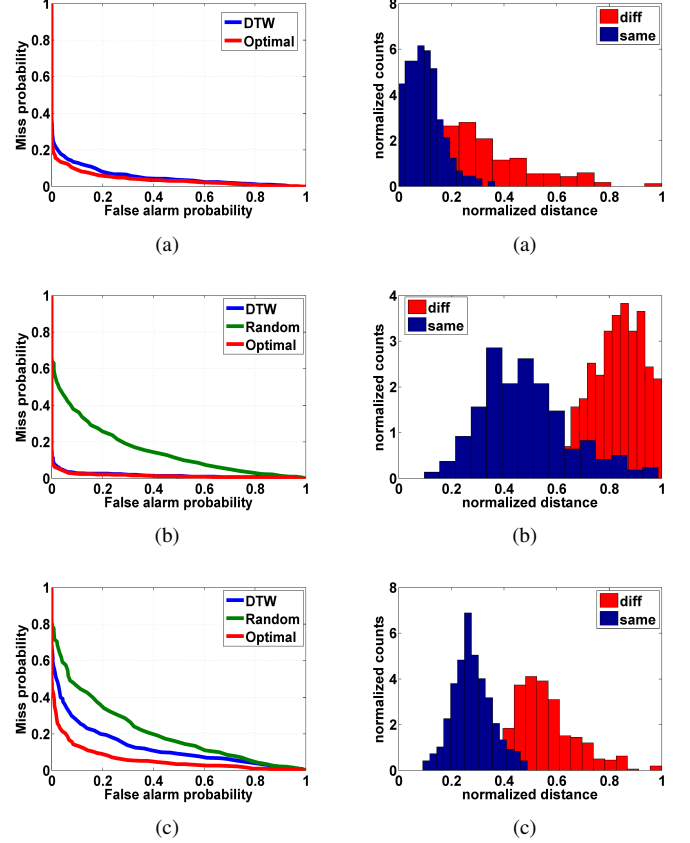


Fig. 4. Benefits of the proposed synchronization method to detection performance under: (a) Spatial attack; (b) Temporal attack; (c) Spatio-temporal attack.

Fig. 5. Histogram of distances between visually different videos (red) and similar videos (blue) (spatial attack): (a) DTW distances; (b) FH; (c) boosted distances.

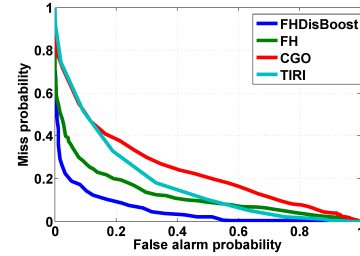


Fig. 6. ROC comparisons against state of the art video hashing techniques: Spatio-temporal attack.

method to get $\mathcal{V}_q^{\text{syn}}$ as described in Algorithm 1, then apply CGO, TIRI and FH to $\mathcal{V}_q^{\text{syn}}$. For FH, we test the performances of using d_{FH} and d_{boost} , respectively. The resulting ROCs are shown in Fig. 6. FH and TIRI are competitive with FH doing slightly better. Finally, FH with distance boosting easily outperforms the alternatives.

4. CONCLUSION

We address the challenge of temporal desynchronization via a novel video hashing framework that involves DTW based synchronization followed by computation of a robust feature vector called flow hash (FH). Further, distance boosting is proposed to capture complementary information in FH and DTW based hash distances which delivers enhanced ROC performance under severe spatio-temporal distortions.

5. REFERENCES

- [1] C. De Roover, C. De Vleeschouwer, F. Lefebvre, and B. Macq, "Robust video hashing based on radial projections of key frames," *Signal Processing, IEEE Transactions on*, vol. 53, no. 10, pp. 4020–4037, 2005.
- [2] B. Coskun, B. Sankur, and N. Memon, "Spatio-temporal transform based video hashing," *Multimedia, IEEE Transactions on*, vol. 8, no. 6, pp. 1190–1208, 2006.
- [3] Sunil Lee and C.D. Yoo, "Robust video fingerprinting for content-based video identification," *Circuits and Systems for Video Technology, IEEE Transactions on*, vol. 18, no. 7, pp. 983–988, 2008.
- [4] M.M. Esmaili, M. Fatourehchi, and R.K. Ward, "A robust and fast video copy detection system using content-based fingerprinting," *Information Forensics and Security, IEEE Transactions on*, vol. 6, no. 1, pp. 213–226, 2011.
- [5] R. Chaudhry, A. Ravichandran, G. Hager, and R. Vidal, "Histograms of oriented optical flow and binet-cauchy kernels on nonlinear dynamical systems for the recognition of human actions," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, 2009, pp. 1932–1939.
- [6] Y.J. Ren, L. O’Gorman, L.J. Wu, Fangzhe Chang, T.L. Wood, and J.R. Zhang, "Authenticating lossy surveillance video," *Information Forensics and Security, IEEE Transactions on*, vol. 8, no. 10, pp. 1678–1687, 2013.
- [7] Jingkuan Song, Yi Yang, Zi Huang, Heng Tao Shen, and Jiebo Luo, "Effective multiple feature hashing for large-scale near-duplicate video retrieval," *Multimedia, IEEE Transactions on*, vol. 15, no. 8, pp. 1997–2008, 2013.
- [8] Xudong Lv and Z.J. Wang, "Compressed binary image hashes based on semisupervised spectral embedding," *Information Forensics and Security, IEEE Transactions on*, vol. 8, no. 11, pp. 1838–1849, 2013.
- [9] Pedro Cano, Eloi Batlle, Ton Kalker, and Jaap Haitsma, "A review of audio fingerprinting," *Journal of VLSI signal processing systems for signal, image and video technology*, vol. 41, no. 3, pp. 271–284, 2005.
- [10] Oztan Harmanci, Mehmet Kucukgoz, and Mehmet K Mihcak, "Temporal synchronization of watermarked video using image hashing," in *Electronic Imaging 2005*. International Society for Optics and Photonics, 2005, pp. 370–380.
- [11] TK Vintsyuk, "Speech discrimination by dynamic programming," *Cybernetics and Systems Analysis*, vol. 4, no. 1, pp. 52–57, 1968.
- [12] Richard Bellman, "An introduction to the theory of dynamic programming," Tech. Rep., DTIC Document, 1953.
- [13] Stan Salvador and Philip Chan, "Toward accurate dynamic time warping in linear time and space," *Intelligent Data Analysis*, vol. 11, no. 5, pp. 561–580, 2007.
- [14] Andrs Bruhn, Joachim Weickert, and Christoph Schnrr, "Lucas/kanade meets horn/schunck: Combining local and global optic flow methods," *International Journal of Computer Vision*, vol. 61, no. 3, pp. 211–231, 2005.
- [15] Yoav Freund and Robert E Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.
- [16] Christopher M Bishop, *Pattern recognition and machine learning*, p. 331, springer New York, 2006.
- [17] Dalwon Jang, Sei-Jin Jang, and Tae-Beom Lim, "Distance combination for content identification system," in *Communications, Signal Processing, and their Applications (ICCSPA), 2013 1st International Conference on*, 2013, pp. 1–6.